It's important to evaluate the performance of the learned hypothesis (in many cases, hypothesis evaluation is even an integral component of the learning method itself).
When evaluating a learned hypothesis we are interested in estimating the accuracy with which it will classify future instances.
Accuracy can be expressed in terms of error:

$$accuracy(h) = 1 - error(h)$$

- **true error**: error rate of the hypothesis over the entire unknown distribution $\mathcal{D}$ of examples
  The true error of hypothesis h with respect to target function f and distribution $\mathcal{D}$ is the probability that h will misclassify an instance drawn at random according to $\mathcal{D}$

$$error_{\mathcal{D}}(h) = \Pr_{x \in \mathcal{D}} [f(x) \neq h(x)]$$

(distribution → $\mathcal{D}$)

- **sample error**: error rate of the hypothesis over the sample of data that is available
  The sample error of h with respect to target function f and data sample S is the proportion of examples that h misclassifies

$$error_S(h) = \frac{1}{n} \sum_{x \in S} \delta(f(x) \neq h(x))$$

where $\delta(f(x) \neq h(x))$ is 1 if $f(x) \neq h(x)$ and 0 otherwise

> we can't compute the true error (we don't know the distribution). We can compute the sample error, but only on a small data sample

How well does the sample error ($error_S(h)$) estimate the true error ($error_{\mathcal{D}}(h)$)?
The learning system should be as accurate as possible → $\forall x \notin S$
Overfitting { $accuracy_S(h)$ is very high; $accuracy_{\mathcal{D}}(h)$ is very low } hypothesis $h \in H$ overfits training data is there is an alternative hypothesis $h' \in H$ such that $error_S(h) < error_S(h')$ and $error_{\mathcal{D}}(h) > error_{\mathcal{D}}(h')$

How to compute the sample error
- partition the dataset $\mathcal{D}$ ($\mathcal{D} = T \cup S$, $T \cap S = \emptyset$, $|T| = 2/3\,|\mathcal{D}|$); the split must be done in a random way
- compute a hypothesis h using training set T
- evaluate $error_S(h) = \frac{1}{n} \sum_{x \in S} \delta(f(x) \neq h(x))$

The sample error is an unbiased estimator for $error_{\mathcal{D}}(h)$

Trade off between training and testing
- having more samples for training and less for testing improves the performance of the model but $error_S(h)$ does not approximate well $error_{\mathcal{D}}(h)$
- having more samples for evaluation and less for training reduces variance of estimation: $error_S(h)$ approximates well $error_{\mathcal{D}}(h)$ but this value may be not satisfactory.

In general, for medium sized datasets: 2/3 is used for training and 1/3 for testing

True comparison between two hypothesis: $d = error_{\mathcal{D}}(h_1) - error_{\mathcal{D}}(h_2)$
Estimator of the comparison between two hypothesis: $\hat{d} = error_{S_1}(h_1) - error_{S_2}(h_2)$
↳ $\hat{d}$ is an unbiased estimator for d iff h_1, h_2, S_1, S_2 are independent from each other

h is the solution of learning algorithm L when using a training set T $\longrightarrow$ $h = L(T)$
Often we are interested in comparing the performance of two learning algorithms L_A and L_B, rather than two specific hypotheses.
$error_S(h)$ is the result of only one experiment and it is not informative on the learning algorithm. We can perform many experiments and compute $error_{S_i}(h)$ for different independent sample data S_i: this is called K-fold Cross Validation method.

K-fold Cross Validation

- partition data set D into K disjoint sets $S_1, S_2, \dots, S_K$ $(|S_i| > 30)$
- for $i = 1, \dots, k$ do
  - use S_i as test set, and the remaining data as training set T_i
    $T_i \longleftarrow \{ D - S_i \}$
    $h_i \longleftarrow L(T_i)$
    $\delta_i \longleftarrow error_{S_i}(h_i)$
- return $error_{L,D} = \frac{1}{K} \sum_{i=1}^{K} \delta_i$ — error is the mean value of all errors computed — $accuracy_{L,D} = 1 - error_{L,D}$

Repeating the process several times we will obtain different hypotheses and different values of $error_{S_i}(h_i)$

Comparing learning algorithms L_A and L_B using K-fold Cross Validation

- partition data set D into K disjoint sets $S_1, S_2, \dots, S_K$ $(|S_i| > 30)$
- for $i = 1, \dots, k$ do
  - use S_i as test set, and the remaining data as training set T_i
    $T_i \longleftarrow \{ D - S_i \}$
    $h_{A_i} \longleftarrow L_A(T_i)$
    $h_{B_i} \longleftarrow L_B(T_i)$
    $\delta_i \longleftarrow error_{S_i}(h_A) - error_{S_i}(h_B)$
- return $\bar{\delta} = \frac{1}{K} \sum_{i=1}^{K} \delta_i$ — error is the mean value of all errors computed — $accuracy_{L,D} = 1 - error_{L,D}$

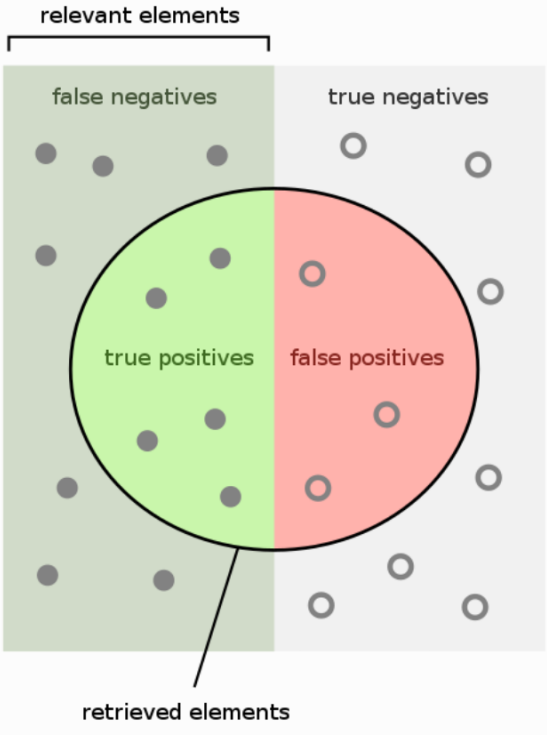
if $\bar{\delta} < 0$ then L_A is better than L_B $(\sim error_{S_i}(h_A) - error_{S_i}(h_B) < 0 \longrightarrow error_{S_i}(h_A) < error_{S_i}(h_B))$
if $\bar{\delta} > 0$ then L_B is better than L_A $(\sim error_{S_i}(h_B) - error_{S_i}(h_A) < 0 \longrightarrow error_{S_i}(h_B) < error_{S_i}(h_A))$

Accuracy only is not enough to assess the performance of a classification method.
Unbalanced data sets are very common in problems related to anomaly detection (e.g. malware analysis, fraud detection, medical tests, ...)

Performance metrics

→ Concept learning = classification problem with two classes

	Predicted class	
True class	YES	NO
YES	True positive	False negative
NO	False positive	True negative

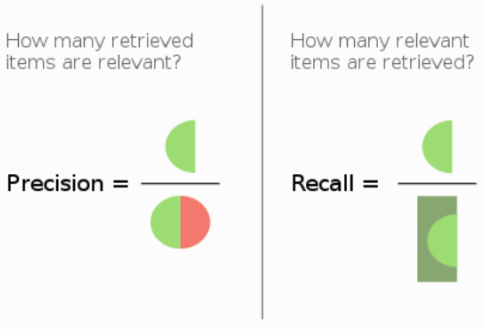


Precision = ability to avoid false positives = $\frac{|TP|}{|TP+FP|}$
= ability to correctly classify positive samples (avoid FP)

Recall = ability to avoid false negatives = $\frac{|TP|}{|TP+FN|}$
precision can be seen as a measure of fidelity (avoid FN)

Precision is the number of true positives compared to the total number of positives identified (true positives + false positives)

Recall is the number of true positives compared to all the existing positives (true positives + false negatives)



→ Classification problem with many (more than two) classes

In a classification problem with many classes, we can compute how many times an instance of class C_i is classified in class C_j.

	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
C_1								
C_2								
C_3								
C_4								
C_5								
C_6								
C_7								
C_8								

confusion matrix

Main diagonal contains the accuracy for each class.
Otside the diagonal, the errors.
Cell C_{ij} contains the times class C_i is classified as class C_j.
e.g. $C_{1,2}$ gives us information on how many times an instance of class C_1 is classified as class C_2

Other performance measures

F_1 - score = $2(Precision \cdot Recall)/(Precision + Recall)$

True Positive Rate TPR (Sensitivity) = $TP/P = TP/(TP+FN)$

True Negative Rate TNR (Specificity) = $TN/N = TN/(TN+FP)$

False Positive Rate FPR = $FP/N = TP/(TN+FP)$

False Negative Rate FNR = $FN/P = FN/(TP+FN)$